CHAPTER 2

Bootstrap CSS

At the core of Bootstrap is a set of basic HTML elements that have been styled to allow for easy enhancement via classes and user styles.

Typography

Starting with typography, Bootstrap uses Helvetica Neue, Helvetica, Arial, and sans-serif in its default font stack. These are all standard fonts and are included as defaults on all major computers. If by chance these fonts don't exist, they fall back to `sans-serif` (the catchall) to tell the browser to use the default font for the browser. All body copy has the `font-size` set at 14 pixels, with the `line-height` set at 20 pixels. The `<p>` tag has a `margin-bottom` of 10 pixels, or half the `line-height`.

Headings

All six standard heading levels have been styled in Bootstrap (see Figure 2-1), with the `<h1>` at 36 pixels tall, and the `<h6>` down to 12 pixels (for reference, default body text is 14 pixels tall). In addition, to add an inline subheading to any of the headings, simply add `<small>` around any of the elements and you will get smaller text in a lighter color. In the case of the `<h1>`, the small text is 24 pixels tall, normal font weight (i.e., not bold), and gray instead of black:

```
h1 small {
    font-size:24px;
        font-weight:normal;
        line-height:1;
        color:#999;
        }
```

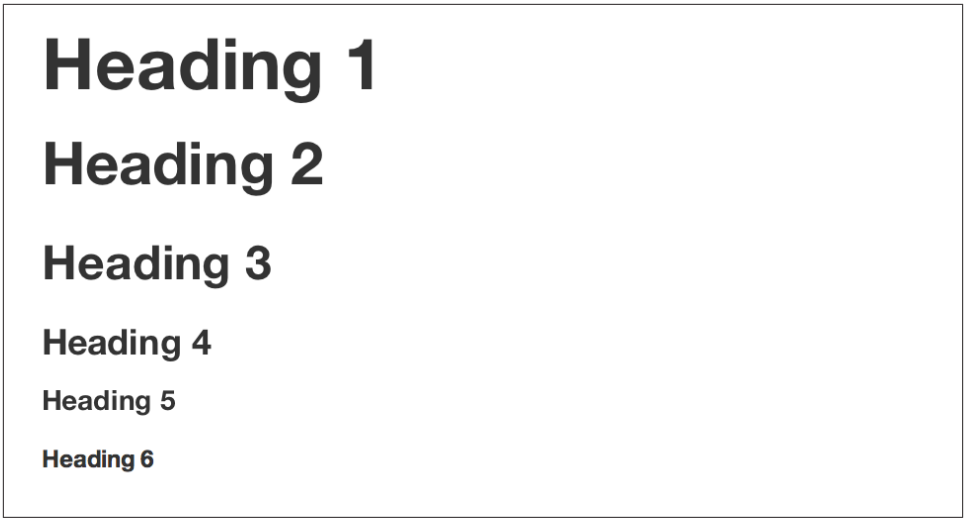


Figure 2-1. Headings

Lead Body Copy

To add some emphasis to a paragraph, add `class="lead"` (see Figure 2-2). This will give you larger font size, lighter weight, and a taller line height. This is generally used for the first few paragraphs in a section, but it can really be used anywhere:

```
<p class="lead">Bacon ipsum dolor sit amet tri-tip pork loin ball tip frankfurter
swine boudin meatloaf shoulder short ribs cow drumstick beef jowl.
Meatball chicken sausage tail, kielbasa strip steak turducken venison prosciutto.
Chuck filet mignon tri-tip ribeye, flank brisket leberkas. Swine
turducken turkey shank, hamburger beef ribs bresaola pastrami venison rump.</p>
```

Lead Example

Bacon ipsum dolor sit amet tri-tip pork loin ball tip frankfurter swine boudin meatloaf shoulder short ribs cow drumstick beef jowl. Meatball chicken sausage tail, kielbasa strip steak turducken venison prosciutto. Chuck filet mignon tri-tip ribeye, flank brisket leberkas. Swine turducken turkey shank, hamburger beef ribs bresaola pastrami venison rump.

Figure 2-2. Lead body copy classes

Emphasis

In addition to using the `<small>` tag within headings, as discussed above, you can also use it with body copy. When `<small>` is applied to body text, the font shrinks to 85% of its original size.

Bold

To add emphasis to text, simply wrap it in a `<strong>` tag. This will add `font-weight:bold;` to the selected text.

Italics

For italics, wrap your content in the `<em>` tag. The term "em" derives from the word "emphasis" and is meant to add stress to your text.

You might be thinking, why not just use the `<b>` or `<i>` tags instead of `<em>` or `<strong>`? In HTML5, `<b>` is meant to highlight words or phrases without conveying additional importance—for example, key terms or names—while `<i>` is mostly for voice, technical terms, internal dialogue, and so on. For more information about the semantic changes to `<b>` and `<i>`, check out W3.org's article.

Emphasis Classes

Along with `<strong>` and `<em>`, Bootstrap offers a few other classes that can be used to provide emphasis (see Figure 2-3). These could be applied to paragraphs or spans:

```
<p class="muted">This content is muted</p>
<p class="text-warning">This content carries a warning class</p>
<p class="text-error">This content carries an error class</p>
<p class="text-info">This content carries an info class</p>
<p class="text-success">This content carries a success class</p>
<p>This content has <em>emphasis</em>, and can be <strong>bold</strong></p>
```

Figure 2-3. Emphasis classes

Abbreviations

The HTML `<abbr>` element provides markup for abbreviations or acronyms, like WWW or HTTP (see Figure 2-4). By marking up abbreviations, you can give useful information to browsers, spell checkers, translation systems, or search engines. Bootstrap styles `<abbr>` elements with a light dotted border along the bottom and reveals the full text on hover (as long as you add that text to the `<abbr>` title attribute):

```
<abbr title="Real Simple Syndication">RSS</abbr>
```

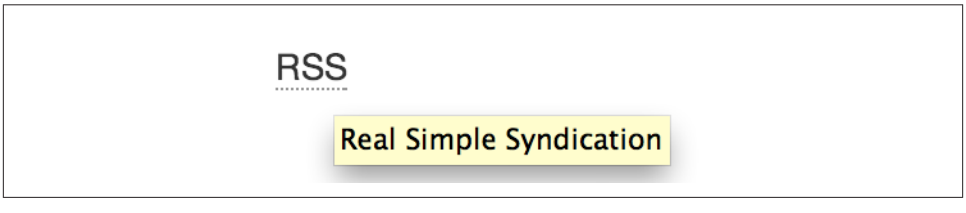


Figure 2-4. Abbreviation example

Add `.initialism` to an `<abbr>` for a slightly smaller font size (see Figure 2-5):

```
<abbr title="rolling on the floor, laughing out loud">That joke had me ROTFLOL
</abbr>
```

Figure 2-5. Another abbreviation example

Addresses

Adding `<address>` elements to your page can help screen readers and search engines locate any physical addresses and phone numbers in the text (see Figure 2-6). It can also be used to mark up email addresses. Since the `<address>` defaults to `display:block;` you'll need to use `<br>` tags to add line breaks to the enclosed address text (e.g., to split the street address and city onto separate lines):

```
<address>
  <strong>O'Reilly Media, Inc.</strong><br>
  1005 Gravenstein HWY North<br>
  Sebastopol, CA 95472<br>
  <abbr title="Phone">P:</abbr> <a href="tel:+17078277000">(707) 827-7000</a>
</address>

<address>
  <strong>Jake Spurlock</strong><br>
  <a href="mailto:#">flast@oreilly.com</a>
</address>
```

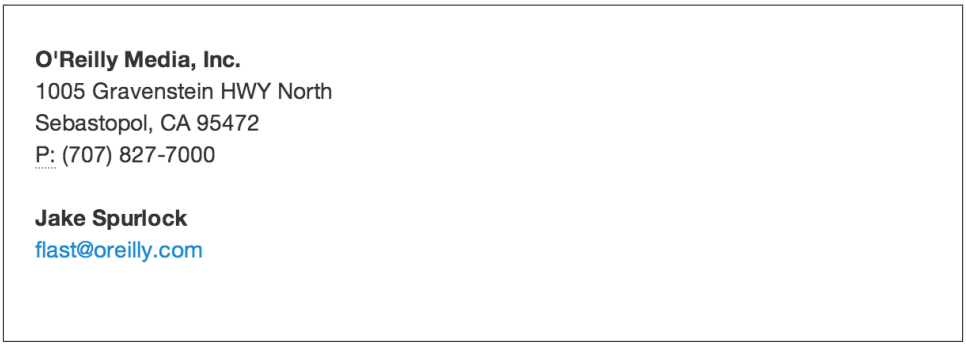


Figure 2-6. Address tag

Blockquotes

To add blocks of quoted text to your document—or for any quotation that you want to set apart from the main text flow—add the `<blockquote>` tag around the text. For best results, and for line breaks, wrap each subsection in a `<p>` tag. Bootstrap's default styling indents the text and adds a thick gray border along the left side. To identify the source

of the quote, add the <small> tag, then add the source's name wrapped in a <cite> tag before closing the </small> tag:

```
<blockquote>
        <p>That this is needed, desperately needed, is indicated by the
    incredible uptake of Bootstrap. I use it in all the server software
    I'm working on. And it shows through in the templating language I'm
    developing, so everyone who uses it will find it's "just there" and
    works, any time you want to do a Bootstrap technique. Nothing to do,
    no libraries to include. It's as if it were part of the hardware.
    Same approach that Apple took with the Mac OS in 1984.</p>
        <small>Developer of RSS, <cite title="Source Title">Dave Winer</cite>
    </small>
</blockquote>
```

When you put it all together, you get something that looks like Figure 2-7.

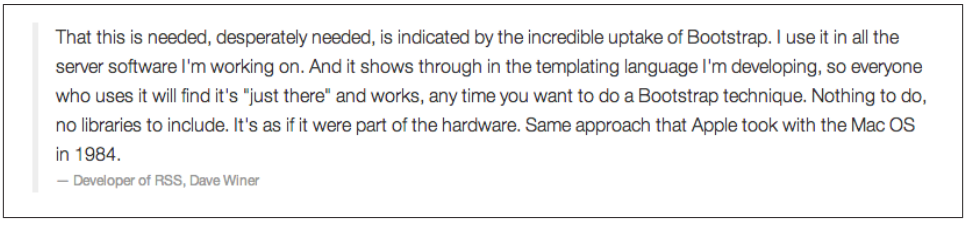


Figure 2-7. Basic blockquote

If you want a <blockquote> with content that is right aligned, add .pull-right to the tag. In addition to the right-aligned text, the entire blockquote is floated to the right. This creates nice pull-quotes in your content, as shown in Figure 2-8.

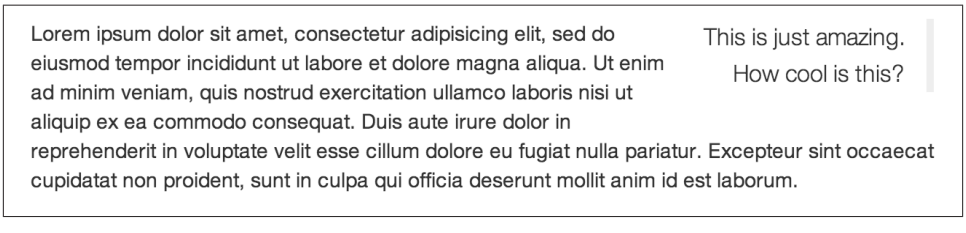


Figure 2-8. Pull right blockquote

Lists

Bootstrap offers support and styling for the three main list types that HTML offers: ordered, unordered, and definition lists. An unordered list is a list that doesn't have any particular order and is traditionally styled with bullets.

Unordered list

If you have an ordered list that you would like to remove the bullets from, add `class="unstyled"` to the opening `<ul>` tag (see Figure 2-9):

```
<h3>Favorite Outdoor Activities</h3>
<ul>
        <li>Backpacking in Yosemite</li>
        <li>Hiking in Arches
                <ul>
                        <li>Delicate Arch</li>
                        <li>Park Avenue</li>
                </ul>
        </li>
        <li>Biking the Flintstones Trail</li>
</ul>
```

Figure 2-9. Unordered list

Personally, I hold a strong aversion to using the `<br>` tag. When I want a single-spaced line break, I place each line in an unstyled, unordered list. For example, if you want a condensed address box, like in Figure 2-6, you could code each line as `<li>`. In my mind, this is a more semantic way to mark up the text.

Ordered list

An ordered list is a list that falls in some sort of sequential order and is prefaced by numbers rather than bullets (see Figure 2-10). This is handy when you want to build a list of numbered items like a task list, guide items, or even a list of comments on a blog post:

```
<h3>Self-Referential Task List</h3>
<ol>
        <li>Turn off the internet.</li>
        <li>Write the book.</li>
        <li>... Profit?</li>
</ol>
```

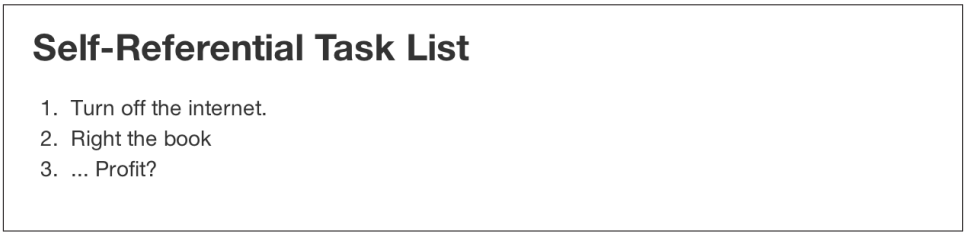


Figure 2-10. Ordered list

Definition list

The third type of list you get with Bootstrap is the definition list. The definition list differs from the ordered and unordered list in that instead of just having a block-level `<li>` element, each list item can consist of both the `<dt>` and the `<dd>` elements. `<dt>` stands for "definition term," and like a dictionary, this is the term (or phrase) that is being defined. Subsequently, the `<dd>` is the definition of the `<dt>`.

A lot of times in markup, you will see people using headings inside an unordered list. This works, but may not be the most semantic way to mark up the text. A better method would be creating a `<dl>` and then styling the `<dt>` and `<dd>` as you would the heading and the text (see Figure 2-11). That being said, Bootstrap offers some clean default styles and an option for a side-by-side layout of each definition:

```
<h3>Common Electronics Parts</h3>
<dl>
        <dt>LED</dt>
        <dd>A light-emitting diode (LED) is a semiconductor light source.</dd>
        <dt>Servo</dt>
        <dd>Servos are small, cheap, mass-produced actuators used for radio
    control and small robotics.</dd>
</dl>
```

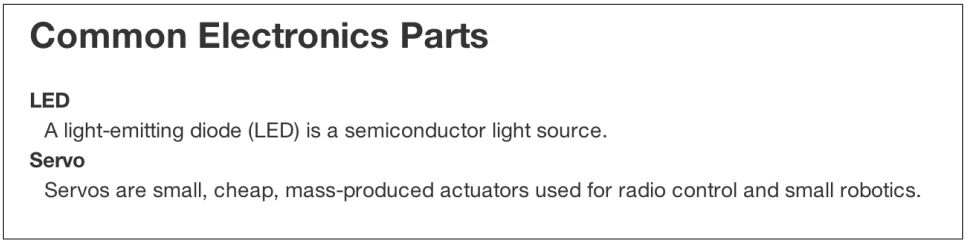


Figure 2-11. Definition list

To change the `<dl>` to a horizontal layout, with the `<dt>` on the left side and the `<dd>` on the right, simply add `class="dl-horizontal"` to the opening tag (see Figure 2-12).

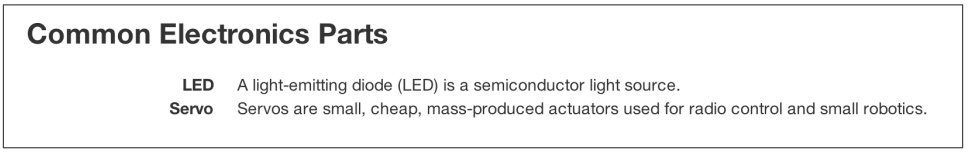


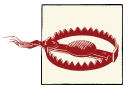
Figure 2-12. Horizontal definition list

Horizontal description lists will truncate terms that are too long to fit in the left column with `text-overflow`. Additionally, in narrower viewports, they will automatically change to the default stacked layout.

Code

There are two different key ways to display code with Bootstrap. The first is the `<code>` tag and the second is the `<pre>` tag. Generally, if you are going to be displaying code inline, you should use the `<code>` tag. But if the code needs to be displayed as a standalone block element or if it has multiple lines, then you should use the `<pre>` tag:

```
<p>Instead of always using divs, in HTML5, you can use new elements like
<code>&lt;section&gt;</code>, <code>&lt;header&gt;</code>, and
<code>&lt;footer&gt;</code>. The html should look something like this:</p>
<pre>
  &lt;article&gt;
    &lt;h1&gt;Article Heading&lt;/h1&gt;
  &lt;/article&gt;
</pre>
```



Make sure that when you use the <pre> and <code> tags, you use the unicode variants for the opening and closing tags: < and >.

Tables

One of my favorite parts of Bootstrap is the nice way that tables are handled. I do a lot of work looking at and building tables, and the clean layout is a great feature that's included in Bootstrap right off the bat. Table 2-1 lists the various elements supported by Bootstrap.

Table 2-1. Table elements supported by Bootstrap

Tag	Description
<table>	Wrapping element for displaying data in a tabular format
<thead>	Container element for table header rows (<tr>) to label table columns
<tbody>	Container element for table rows (<tr>) in the body of the table
<tr>	Container element for a set of table cells (<td> or <th>) that appears on a single row
<td>	Default table cell
<th>	Special table cell for column (or row, depending on scope and placement) labels. Must be used within a <thead>
<caption>	Description or summary of what the table holds, especially useful for screen readers

If you want a nice, basic table style with just some light padding and horizontal dividers, add the base class of `.table` to any table (see Figure 2-13). The basic layout has a top border on all of the <td> elements:

```
<table class="table">
  <caption>...</caption>
  <thead>
    <tr>
      <th>...</th>
      <th>...</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>...</td>
      <td>...</td>
    </tr>
  </tbody>
</table>
```

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-13. Basic table class

Optional Table Classes

Along with the base table markup and the `.table` class, there are a few additional classes that you can use to style the markup. These four classes are: `.table-striped`, `.table-bordered`, `.table-hover`, and `.table-condensed`.

Striped table

By adding the `.table-striped` class, you will get stripes on rows within the `<tbody>` (see Figure 2-14). This is done via the CSS `:nth-child` selector, which is not available on Internet Explorer 7–8.

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-14. Striped table class

Bordered table

If you add the `.table-bordered` class, you will get borders surrounding every element and rounded corners around the entire table, as shown in Figure 2-15.

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-15. Bordered table class

Hover table

Figure 2-16 shows the `.table-hover` class. A light gray background will be added to rows while the cursor hovers over them.

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-16. Hover table class

Condensed table

If you add the `.table-condensed` class, as shown in Figure 2-17, row padding is cut in half to condense the table. This is useful if you want denser information.

Name	Phone Number	Rank
Kyle West	707-827-7001	Eagle
Davey Preston	707-827-7003	Eagle
Taylor Lemmon	707-827-7005	Eagle

Figure 2-17. Condensed table class

Table Row Classes

The classes shown in Table 2-2 will allow you to change the background color of your rows (see Figure 2-18).

Table 2-2. Optional table row classes

Class	Description	Background color
`.success`	Indicates a successful or positive action.	Green
`.error`	Indicates a dangerous or potentially negative action.	Red
`.warning`	Indicates a warning that might need attention.	Yellow
`.info`	Used as an alternative to the default styles.	Blue

#	Product	Payment Taken	Status
1	TB - Monthly	01/04/2012	Approved
2	TB - Monthly	02/04/2012	Declined
3	TB - Monthly	03/04/2012	Pending
4	TB - Monthly	04/04/2012	Call in to confirm

Figure 2-18. Table row classes

Forms

Another one of the highlights of using Bootstrap is the ability to create forms with ease. As a web developer, styling forms is one of my least favorite tasks. Bootstrap makes it easy with the simple HTML markup and extended classes for different styles of forms.

The basic form structure comes with Bootstrap; there is no need to add any extra helper classes (see Figure 2-19). If you use the placeholder, keep in mind that it is only supported in newer browsers. In older browsers, no placeholder text will be displayed:

```
<form>
        <fieldset>
                <legend>Legend</legend>
                        <label for="name">Label name</label>
                        <input type="text" id="name"
                   placeholder="Type something…">
                        <span class="help-block">Example block-level help
                         text here.</span>
                        <label class="checkbox" for="checkbox">
                                <input type="checkbox" id="checkbox">
                Check me out
                        </label>
                <button type="submit" class="btn">Submit</button>
        </fieldset>
</form>
```

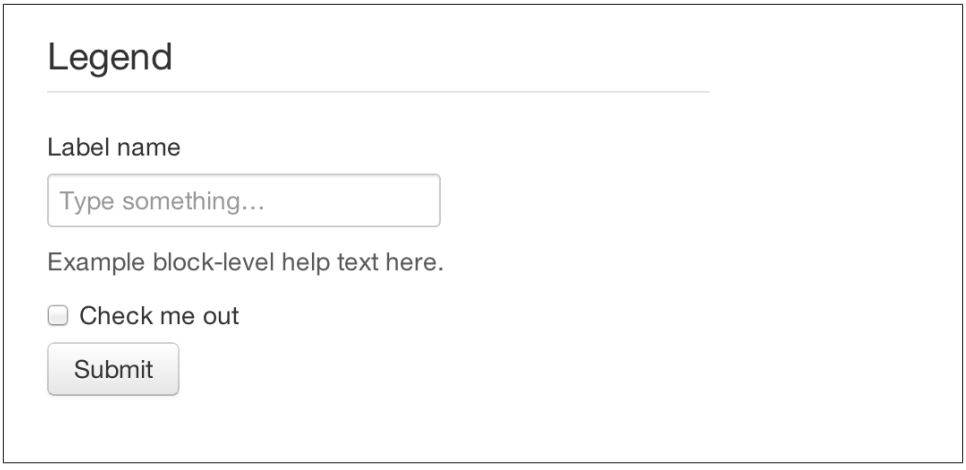


Figure 2-19. Basic form

Optional Form Layouts

With a few helper classes, you can dynamically update the layout of your form. Bootstrap comes with a few preset styles to choose from.

Search form

Add `.form-search` to the `<form>` tag, and then add `.search-query` to the `<input>` for an input box with rounded corners and an inline submit button (see Figure 2-20):

```
<form class="form-search">
  <input type="text" class="input-medium search-query">
  <button type="submit" class="btn">Search</button>
</form>
```

Figure 2-20. Search form

Inline form

To create a form where all of the elements are inline and labels are alongside, add the class `.form-inline` to the `<form>` tag (see Figure 2-21). To have the label and the input on the same line, use this inline form code:

```
<form class="form-inline">
        <input type="text" class="input-small" placeholder="Email">
        <input type="password" class="input-small" placeholder="Password">
```

```
        <label class="checkbox">
                <input type="checkbox"> Remember me
        </label>
        <button type="submit" class="btn">Sign in</button>
</form>
```

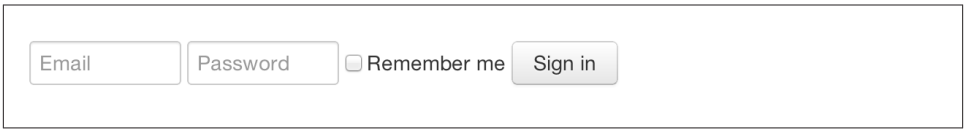


Figure 2-21. Inline form

Horizontal form

Bootstrap also comes with a prebaked horizontal form; this one stands apart from the others not only in the amount of markup, but also in the presentation of the form. Traditionally you'd use a table to get a form layout like the one shown in Figure 2-22, but Bootstrap manages to do it without using tables. Even better, if you're using the responsive CSS, the horizontal form will automatically adapt to smaller layouts by stacking the controls vertically.

To create a form that uses the horizontal layout, do the following:

- Add a class of `.form-horizontal` to the parent `<form>` element.
- Wrap labels and controls in a `<div>` with class `.control-group`.
- Add a class of `.control-label` to the labels.
- Wrap any associated controls in a `<div>` with class `.controls` for proper alignment.

Figure 2-22. Horizontal form

```
<form class="form-horizontal">
  <div class="control-group">
    <label class="control-label" for="inputEmail">Email</label>
```

```
    <div class="controls">
      <input type="text" id="inputEmail" placeholder="Email">
    </div>
  </div>
  <div class="control-group">
    <label class="control-label" for="inputPassword">Password</label>
    <div class="controls">
      <input type="password" id="inputPassword" placeholder="Password">
    </div>
  </div>
  <div class="control-group">
    <div class="controls">
      <label class="checkbox">
        <input type="checkbox"> Remember me
      </label>
      <button type="submit" class="btn">Sign in</button>
    </div>
  </div>
</form>
```

Supported Form Controls

Bootstrap natively supports the most common form controls. Chief among them are `input`, `textarea`, `checkbox`, `radio`, and `select`.

Inputs

The most common form text field is the input—this is where users will enter most of the essential form data (see Figure 2-23). Bootstrap offers support for all native HTML5 input types: `text`, `password`, `datetime`, `datetime-local`, `date`, `month`, `time`, `week`, `number`, `email`, `URL`, `search`, `tel`, and `color`:

```
<input type="text" placeholder="Text input">
```

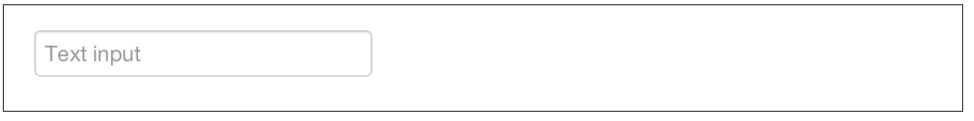
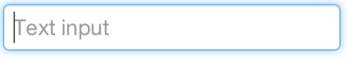


Figure 2-23. Input

Both `input` and `textarea` default to a nice blue glow when in the `:active` state.



Textarea

The `textarea` is used when you need multiple lines of input (see Figure 2-24). You'll find you mainly modify the `rows` attribute, changing it to the number of rows that you need to support (fewer rows = smaller box, more rows = bigger box):

```
<textarea rows="3"></textarea>
```

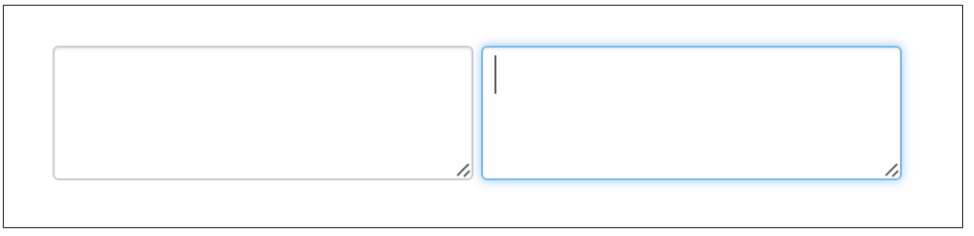


Figure 2-24. Both the :active default and the textarea

Checkboxes and radio buttons

Checkboxes and radio buttons are great when you want users to choose from a list of preset options (see Figure 2-25). When building a form, use `checkbox` if you want the user to select any number of options from a list. Use `radio` if you want to limit him to just one selection:

```
<label class="checkbox">
  <input type="checkbox" value="">
  Option one is this and that—be sure to include why it's great.
</label>

<label class="radio">
  <input type="radio" name="optionsRadios" id="optionsRadios1" value="option1"
  checked>
  Option one is this and that—be sure to include why it's great.
</label>
<label class="radio">
  <input type="radio" name="optionsRadios" id="optionsRadios2" value="option2">
  Option two can be something else, and selecting it will deselect option one
</label>
```

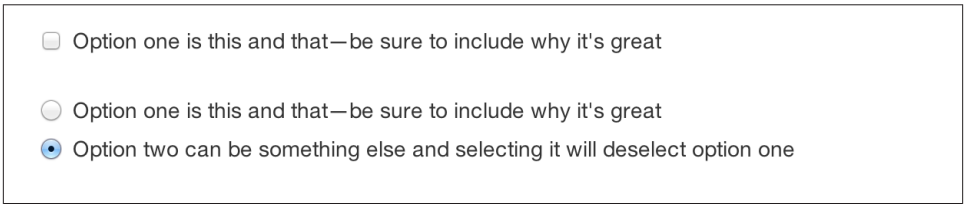


Figure 2-25. Checkbox and radio buttons

If you want multiple checkboxes to appear on the same line together, add the `.inline` class to a series of checkboxes or radio buttons (see Figure 2-26):

```
<label for="option1" class="checkbox inline">
  <input id="option1" type="checkbox" id="inlineCheckbox1" value="option1"> 1
</label>
<label for="option2" class="checkbox inline">
  <input id="option2" type="checkbox" id="inlineCheckbox2" value="option2"> 2
</label>
<label for="option3" class="checkbox inline">
  <input id="option3" type="checkbox" id="inlineCheckbox3" value="option3"> 3
</label>
```

Figure 2-26. Inline checkboxes

Selects

A select is used when you want to allow the user to pick from multiple options, but by default it only allows one (see Figure 2-27). It's best to use `<select>` for list options with which the user is familiar, such as states or numbers. Use `multiple="multiple"` to allow the user to select more than one option. If you only want the user to choose one option, use `type="radio"`:

```
<select>
  <option>1</option>
  <option>2</option>
  <option>3</option>
  <option>4</option>
  <option>5</option>
</select>

<select multiple="multiple">
  <option>1</option>
  <option>2</option>
  <option>3</option>
  <option>4</option>
  <option>5</option>
</select>
```

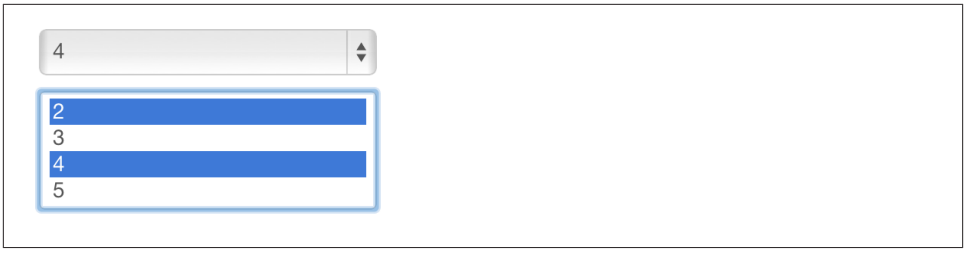


Figure 2-27. Select

Extended Form Controls

In addition to the basic form controls listed in the previous section, Bootstrap offers a few other form components to complement the standard HTML form elements; for example, it lets you easily prepend and append content to inputs.

Prepended and appended inputs

By adding prepended and appended content to an input field, you can add common elements to the user's input (see Figure 2-28). For example, you can add the dollar symbol, the `@` for a Twitter username, or anything else that might be common for your application interface. To add extra content before the user input, wrap the prepended input in a `<div>` with class `.input-prepend`. To append input, use the class `.input-append`. Then, within that same `<div>`, place your extra content inside a `<span>` with an `.add-on` class, and place the `<span>` either before or after the `<input>` element:

```
<div class="input-prepend">
  <span class="add-on">@</span>
  <input class="span2" id="prependedInput" type="text" placeholder="Username">
</div>
<div class="input-append">
  <input class="span2" id="appendedInput" type="text">
  <span class="add-on">.00</span>
</div>
```

Figure 2-28. Prepend and append

If you combine both of them, you simply need to add both the `.input-prepend` and `.input-append` classes to the parent `<div>` (see Figure 2-29):

```
<div class="input-prepend input-append">
  <span class="add-on">$</span>
  <input class="span2" id="appendedPrependedInput" type="text">
  <span class="add-on">.00</span>
</div>
```

Figure 2-29. Using both the append and prepend

Rather than using a `<span>`, you can instead use `<button>` with a class of `.btn` to attach (surprise!) a button or two to the input (see Figure 2-30):

```
<div class="input-append">
  <input class="span2" id="appendedInputButtons" type="text">
  <button class="btn" type="button">Search</button>
  <button class="btn" type="button">Options</button>
</div>
```

Figure 2-30. Attach multiple buttons to an input

If you are appending a button to a search form, you will get the same nice rounded corners that you would expect (see Figure 2-31):

```
<form class="form-search">
  <div class="input-append">
    <input type="text" class="span2 search-query">
    <button type="submit" class="btn">Search</button>
  </div>
  <div class="input-prepend">
    <button type="submit" class="btn">Search</button>
    <input type="text" class="span2 search-query">
  </div>
</form>
```

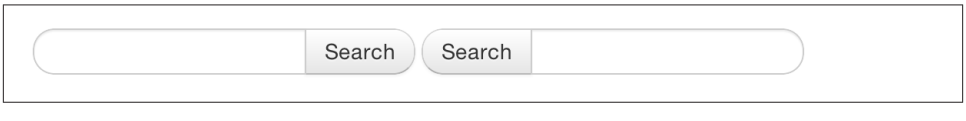


Figure 2-31. Append button to search form

Form Control Sizing

With the default grid system that is inherent in Bootstrap, you can use the `.span*` system for sizing form controls. In addition to the span column-sizing method, you can also use a handful of classes that take a relative approach to sizing. If you want the input to act as a block-level element, you can add `.input-block-level` and it will be the full width of the container element, as shown in Figure 2-32:

```
<input class="input-block-level" type="text" placeholder=".input-block-level">
```

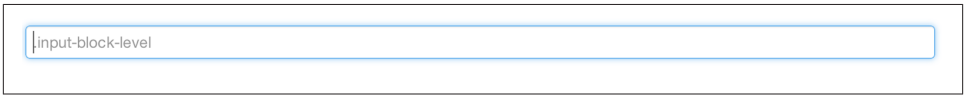


Figure 2-32. Block-level input

Relative input controls

In addition to using `.span*` for input sizing, you can also use a few different class names (see Figure 2-33):

```
<input class="input-mini" type="text" placeholder=".input-mini">
<input class="input-small" type="text" placeholder=".input-small">
<input class="input-medium" type="text" placeholder=".input-medium">
<input class="input-large" type="text" placeholder=".input-large">
<input class="input-xlarge" type="text" placeholder=".input-xlarge">
<input class="input-xxlarge" type="text" placeholder=".input-xxlarge">
```

Figure 2-33. Relative input controls

In future versions of Bootstrap, these input classes will be altered to match the button sizes. For example, `.input-large` will increase the padding and font size of an input.

Grid sizing

You can use any `.span` from `.span1` to `.span12` for form control sizing (see Figure 2-34):

```
<input class="span1" type="text" placeholder=".span1">
<input class="span2" type="text" placeholder=".span2">
<input class="span3" type="text" placeholder=".span3">
<select class="span1">
  ...
</select>
<select class="span2">
  ...
</select>
<select class="span3">
  ...
</select>
```

Figure 2-34. Span-sized inputs

If you want to use multiple inputs on a line, simply use the `.controls-row` modifier class to apply the proper spacing (see Figure 2-35). It floats the inputs to collapse the white space; sets the correct margins; and, like the `.row` class, clears the float:

```
<div class="controls">
  <input class="span5" type="text" placeholder=".span5">
</div>
<div class="controls controls-row">
  <input class="span4" type="text" placeholder=".span4">
  <input class="span1" type="text" placeholder=".span1">
```

```
</div>
...
```

Figure 2-35. Control row

Uneditable text

If you want to present a form control without allowing the user to edit the input, simply add the class `.uneditable-input` (see Figure 2-36):

```
<span class="input-xlarge uneditable-input">Some value here</span>
```

Figure 2-36. Uneditable input

Form actions

When you place the form actions at the bottom of a `.horizontal-form`, the inputs will correctly line up with the floated form controls (see Figure 2-37):

```
<div class="form-actions">
  <button type="submit" class="btn btn-primary">Save changes</button>
  <button type="button" class="btn">Cancel</button>
</div>
```

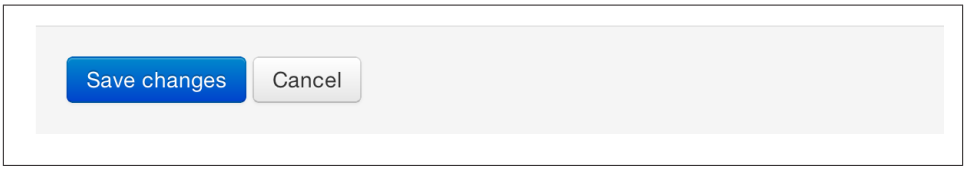


Figure 2-37. Form controls

Help text

Bootstrap form controls can have either block or inline text that flows with the inputs (see Figure 2-38):

```
<input type="text"><span class="help-inline">Inline help text</span>
```

Figure 2-38. Inline help

To add a full width block of content, use the `.help-block` after the `<input>` (see Figure 2-39):

```
<input type="text"><span class="help-block">A longer block of help text that
breaks onto a new line and may extend beyond one line.</span>
```

Figure 2-39. Block help

Form Control States

In addition to the `:focus` state, Bootstrap offers styling for disabled inputs and classes for form validation.

Input focus

When an input receives `:focus` (i.e., a user clicks into the input or tabs onto it), the outline of the input is removed and a `box-shadow` is applied. I remember the first time that I saw this on Twitter's site; it blew me away, and I had to dig into the code to see how they did it. In WebKit, this is accomplished in the following manner:

```
input {
  -webkit-box-shadow: inset 0 1px 1px rgba(0, 0, 0, 0.075);
  -webkit-transition: box-shadow linear 0.2s;
}

input:focus {
  -webkit-box-shadow: inset 0 1px 1px rgba(0, 0, 0, 0.075), 0 0 8px
                      rgba(82, 168, 236, 0.6);
}
```

The <input> has a small inset box-shadow, which gives the appearance that the input sits lower than the page (see Figure 2-40). When :focus is applied, an 8px light blue border appears. The webkit-transition tells the browser to apply the effect in a linear manner over 0.2 seconds:

```
<input class="input-xlarge" id="focusedInput" type="text"
              value="This is focused...">
```

Figure 2-40. Focused input

Nice and subtle; a great effect.

Disabled input

If you need to disable an input, simply adding the disabled attribute will not only disable it; it will also change the styling and the mouse cursor when the cursor hovers over the element (see Figure 2-41):

```
<input class="input-xlarge" id="disabledInput" type="text"
              placeholder="Disabled input here..." disabled>
```

Figure 2-41. Disabled input

Validation states

Bootstrap includes validation styles for error, warning, info, and success messages (see Figure 2-42). To use, simply add the appropriate class to the surrounding .control-group:

```
<div class="control-group warning">
  <label class="control-label" for="inputWarning">Input with warning</label>
  <div class="controls">
    <input type="text" id="inputWarning">
    <span class="help-inline">Something may have gone wrong</span>
  </div>
</div>
<div class="control-group error">
  <label class="control-label" for="inputError">Input with error</label>
  <div class="controls">
    <input type="text" id="inputError">
```

```
    <span class="help-inline">Please correct the error</span>
  </div>
</div>
<div class="control-group success">
  <label class="control-label" for="inputSuccess">Input with success</label>
  <div class="controls">
    <input type="text" id="inputSuccess">
    <span class="help-inline">Woohoo!</span>
  </div>
</div>
```

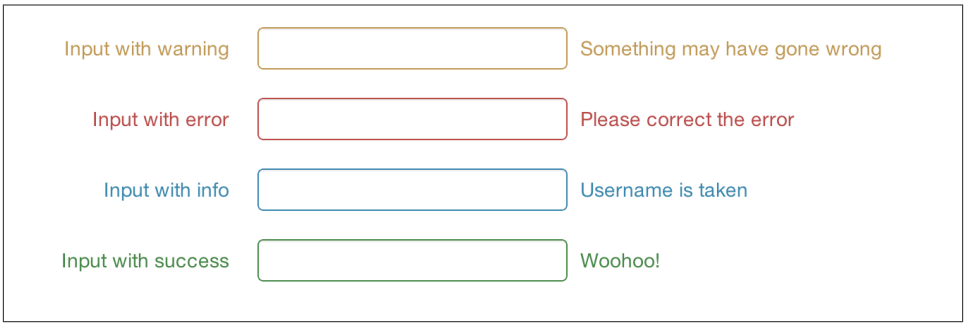


Figure 2-42. Validation states

Buttons

One of my favorite features of Bootstrap is the way that buttons are styled. Dave Winer, inventor of RSS and big fan of Bootstrap, has this to say about it:

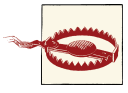
> That this is needed, desperately needed, is indicated by the incredible uptake of Bootstrap. I use it in all the server software I'm working on. And it shows through in the templating language I'm developing, so everyone who uses it will find it's "just there" and works, any time you want to do a Bootstrap technique. Nothing to do, no libraries to include. It's as if it were part of the hardware. Same approach that Apple took with the Mac OS in 1984.
>
> — Dave Winer
> *scripting.com*

I like to think that Bootstrap is unifying the Web and allowing a unified experience of what an interface can look like across the Web. With the advent of Bootstrap, you can usually spot the sites that have adopted it by the buttons that they use. A grid layout and many of the other features fade into the background, but buttons, forms, and other unifying elements are a key part of Bootstrap. When I come across a site that is using Bootstrap, I want to give a high five to the webmaster at that domain, since he probably "just gets it." I felt the same way a few years ago whenever I saw `wp-content` in the HTML of sites that I visited.

Now, buttons and links can all look alike with Bootstrap. Anything that is given a class of `.btn` will inherit the default look of a gray button with rounded corners. However, you can add color to the buttons by adding extra classes (see Table 2-3).

Table 2-3. Button color examples

Buttons	Class	Description
Default	`btn`	Standard gray button with gradient
Primary	`btn btn-primary`	Provides extra visual weight and identifies the primary action in a set of buttons (blue)
Info	`btn btn-info`	Used as an alternative to the default styles (light blue)
Success	`btn-success`	Indicates a successful or positive action (green)
Warning	`btn btn-warning`	Indicates caution should be taken with this action (orange)
Danger	`btn btn-danger`	Indicates a dangerous or potentially negative action (red)
Inverse	`btn btn-inverse`	Alternate dark-gray button, not tied to a semantic action or use
Link	`btn btn-link`	De-emphasizes a button by making it look like a link while maintaining button behavior



There are issues with buttons not appearing in Internet Explorer 9 because it doesn't crop background gradients on rounded corners. Also, Internet Explorer doesn't work well with disabled button elements. The rendered text is gray with a nasty text shadow that hasn't been fixed.

Button Sizes

If you need larger or smaller buttons, simply add `.btn-large`, `.btn-small`, or `.btn-mini` to links or buttons (see Figure 2-43):

```
<p>
  <button class="btn btn-large btn-primary" type="button">Large button</button>
  <button class="btn btn-large" type="button">Large button</button>
</p>
<p>
  <button class="btn btn-primary" type="button">Default button</button>
  <button class="btn" type="button">Default button</button>
</p>
<p>
  <button class="btn btn-small btn-primary" type="button">Small button</button>
```

```
  <button class="btn btn-small" type="button">Small button</button>
</p>
<p>
  <button class="btn btn-mini btn-primary" type="button">Mini button</button>
  <button class="btn btn-mini" type="button">Mini button</button>
</p>
```

Figure 2-43. Different button sizes

If you want to create buttons that display like a block-level element, simply add the `.btn-block` class (see Figure 2-44). These buttons will display at 100% width:

```
<button class="btn btn-large btn-block btn-primary" type="button">Block-
level button</button>
<button class="btn btn-large btn-block" type="button">Block-level button</button>
```

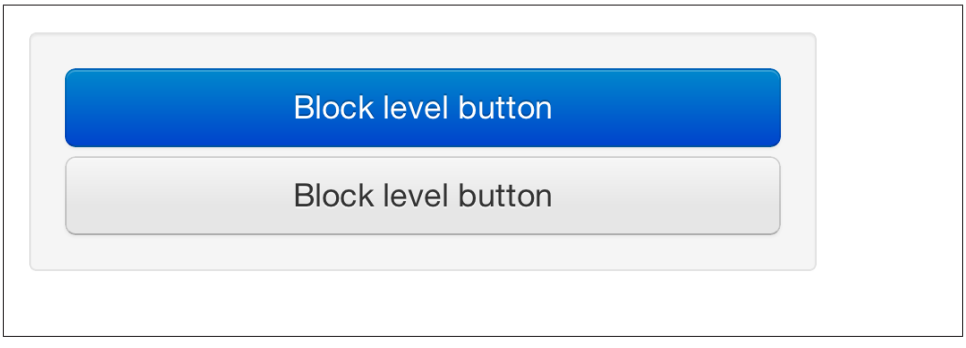


Figure 2-44. Block-level button

Disabled Button Styling

For anchor elements, simply add the class of `.disabled` to the tag and the link will fade in color, and lose the gradient (see Figure 2-45):

```
<a href="#" class="btn btn-large btn-primary disabled">Primary link</a>
<a href="#" class="btn btn-large disabled">Link</a>
```

Figure 2-45. Disabled link

The `.disabled` class is being used much like the `.active` class. So, there's no `.btn` prefix, and remember, this is only for looks. You will need to use some JavaScript to actually disable the link.

For a button, simply add the `disabled` attribute to the button (see Figure 2-46). This will actually disable the button, so JavaScript is not directly needed:

```
<button type="button" class="btn btn-large btn-primary disabled"
disabled="disabled">Primary button</button>
<button type="button" class="btn btn-large" disabled>Button</button>
```

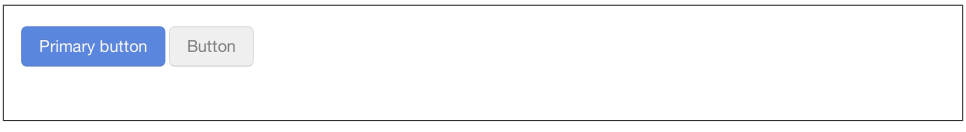


Figure 2-46. Disabled button

Images

Images have three classes (see Figure 2-47) that can be used to apply some simple styles: `.img-rounded` adds `border-radius:6px` to give the image rounded corners, `.img-circle` makes the entire image round by adding `border-radius: 500px`, and `.img-polaroid` adds a bit of padding and a gray border:

```
<img src="..." class="img-rounded">
<img src="..." class="img-circle">
<img src="..." class="img-polaroid">
```

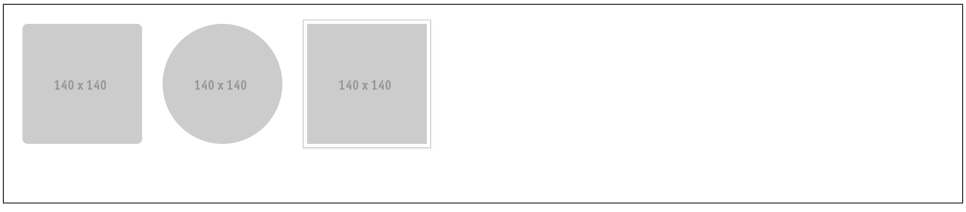


Figure 2-47. Images

Icons

Bootstrap bundles 140 icons into one sprite that can be used with buttons, links, navigation, and form fields. The icons are provided by GLYPHICONS; see Figure 2-48.

Figure 2-48. Icons by GLYPHICONS

GLYPHICONS Attribution

Users of Bootstrap are fortunate to use the GLYPHICONS for free on Bootstrap projects. The developers have asked that you link back to GLYPHICONS when practical.

> GLYPHICONS Halflings are normally not available for free, but an arrangement between Bootstrap and the GLYPHICONS creators have made this possible at no cost to you as developers. As a thank you, we ask you to include an optional link back to GLYPHICONS whenever practical.
>
> — Bootstrap Documentation
> *http://getbootstrap.com*

Usage

To use the icons, simply use an `<i>` tag with the namespaced `.icon-` class. For example, if you want to use the edit icon, you add the `.icon-edit` class to the `<i>` tag:

```
<i class="icon-edit"></i>
```

If you want to use the white icon, simply add the `.icon-white` class to the tag:

```
<i class="icon-edit icon-white"></i>
```

Button groups

By using button groups combined with icons, you can create nice interface elements with minimal markup (see Figure 2-49):

```
<div class="btn-toolbar">
  <div class="btn-group">
    <a class="btn" href="#"><i class="icon-align-left"></i></a>
    <a class="btn" href="#"><i class="icon-align-center"></i></a>
    <a class="btn" href="#"><i class="icon-align-right"></i></a>
    <a class="btn" href="#"><i class="icon-align-justify"></i></a>
  </div>
</div>
```

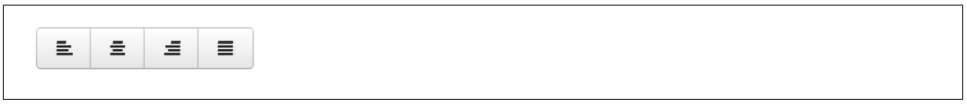


Figure 2-49. Button groups

Navigation

When you are using icons next to a string of text, make sure to add a space to provide the proper alignment of the image (see Figure 2-50). Navigation code will be covered further in the next chapter.

```
<ul class="nav nav-list">
  <li class="active"><a href="#"><i class="icon-home icon-white"></i>
  Home</a></li>
  <li><a href="#"><i class="icon-book"></i> Library</a></li>
  <li><a href="#"><i class="icon-pencil"></i> Applications</a></li>
  <li><a href="#"><i class="i"></i> Misc</a></li>
</ul>
```

Figure 2-50. Basic navigation list